2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt – CCDWN Workshop)

Stochastic D2D Caching with Energy Harvesting Nodes

Homa Nikbakht[1], Sarah Kamel[1], Michèle Wigger[1] and Aylin Yener[2]
[1] LTCI, Télécom Paris, France, {homa.nikbakht, sarah.kamel,michele.wigger}@telecom-paristech.fr
[2] Wireless Communications and Networking Laboratory, The Pennsylvania State University, yener@ee.psu.edu

***Abstract*—Consider a stochastic wireless device-to-device (D2D) caching network with nodes that are harvesting energy from external sources at random times. Each node is equipped with a cache memory, where the node prefetches maximum distance separable (MDS) coded packets of the files from a given library. When a node requests a file from this library, neighbouring nodes are asked to send the relevant missing subfiles over noisy channels. This work presents different selection strategies to determine which neighbouring nodes should transmit which missing subfiles. The strategies can roughly be divided into three categories: *sequential strategies* where transmission stops when the requesting node has correctly decoded enough subfiles; *coordinated strategies* where the requesting node is informed about the other nodes' cache contents and centrally decides which node should send which file; and *adaptive strategies* where the requesting node sequentially decides on which files should be sent in function of the subfiles that it previously decoded correctly. Our numerical simulations show that at moderate energy levels or when there are many file requests, sequential strategies perform significantly worse than coordinated or adaptive strategies. On the other hand, at high energy levels sequential strategies (or even completely decentralized strategies) perform as well or even better. These latter strategies should thus be prefered as they come with less synchronization overhead and delay. The same applies for environments with only few transmission errors (i.e., in high quality channels).**

I. INTRODUCTION

Self-powered, energy harvesting nodes are expected to be a promising solution for future networks with growing data traffic [1]. Energy harvesting enables communications without human intervention for energy replenishment. However, the amount of energy that a node possesses at a given time depends on the energy arrivals and the previously consumed energy. This can lead to energy outage, where a node wishes to transmit information but does not have enough energy to do so. Judicious power allocation and scheduling algorithms can avoid such outage events allowing perpetual operation. References [2] and [3] have studied the problem of optimal transmission scheduling under intermittent energy arrivals that can be stored in a finite sized energy storage unit. Several extensions of this framework including multiterminal setups, finite data buffers, storage losses and online polices with casual knowledge of energy arrivals have also been studied extensively, see e.g. [4]–[7].

Another promising approach to reduce energy consumption, is to prefetch chunks of data in the cache memories directly at the nodes [8]–[11]. More specifically, each node fills its cache memory with coded or uncoded packets of data, before deciding which data it wishes to learn, e.g., which movie it wishes to download. Caching enables saving of resources during the times of data requests [8]. In [9], energy-harvesting small cells fetch popular contents from a macro base station and then consume the harvested energy to push the content to the users proactively. Simulation results in [10] indicate that increasing cache sizes and/or energy harvesting capabilities increases energy availability in such small base stations and improves throughput. Reference [11] proposes mechanisms to decide whether base stations should accept an incoming content request or not, and to determine whether the content should be cached or not.

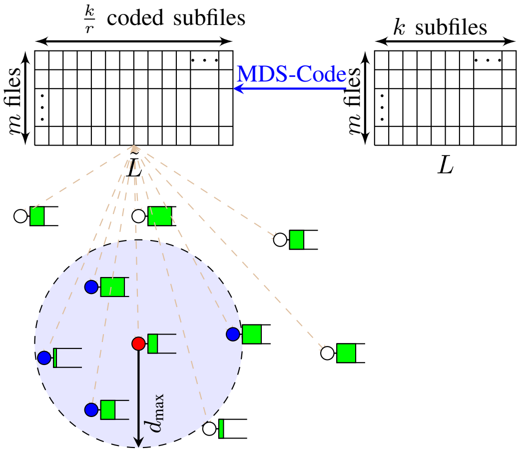

Fig. 1: A stochastic wireless D2D caching network with energy harvesting nodes. The red node is a demanding node and only nodes that are within a distance of $d_{\max}$ of this node send the relevant missing subfiles.

In this work, we consider a decentralized caching network with energy harvesting transmissions. We do not have a central controller that designs the cache placement or schedules transmissions. The energy harvesting nodes that are equipped with cache memories prefetch coded packets from a library. File demands are satisfied locally via D2D communications, by leveraging the collective content of caches memories of nearby nodes. Each node can send any of the subfiles in its cache memory, subject to its available energy to another node in range. Energy and file requests arrive at time-instances indicated by stochastic counting processes of different rates.

ISBN 978-3-903176-29-4 © 2020 IFIP

In this setup, we study strategies on how to address a file request, i.e., we describe different mechanisms to decide which node should send which subfile. Depending on the strategy, this decision is made in a distributed manner at the various nodes, or in a coordinated fashion by the node that is requesting the file. In some of the strategies, the decision is even allowed to depend on the subfiles that have previously been decoded without error.

Our numerical results demonstrate that coordinated strategies perform much better than the uncoordinated strategies when energy is a limiting resource, e.g., when energy harvesting rates are low or when the number of file requests is high with respect to the harvested energy. By contrast, when energy is abundant in the system, simpler uncoordinated strategies with less synchronization overhead and less delay suffice. We also observe that when there are only few transmission errors, it is advantageous to use strategies that stop transmitting once the requesting node has attained all the requested subfiles.

II. PROBLEM SETUP

Consider a network composed of a set of nodes $\mathcal{N}$, which are generated according to a Poisson Point Process with rate $\lambda > 0$. The nodes harvest energy from external sources and have cache memories where they prefetch contents from a common library $\mathcal{L}$ of m files:

$$\mathcal{L} := \{f_1, \ldots, f_m\}. \tag{1}$$

Each file f_i, for $i \in \{1, \ldots, m\}$, consists of F independent and identically distributed (i.i.d.) bits. To diversify the prefetched content, each file f_i is divided into $k > 0$ subfiles $f_i^{(1)}, \ldots, f_i^{(k)}$ of equal size F/k bits, and the subfiles are encoded into $\frac{k}{r}$ encoded subfiles $\tilde{f}_i^{(1)}, \ldots, \tilde{f}_i^{(\frac{k}{r})}$ by means of a $(k, \frac{k}{r})$ MDS code.[1] Thus, for any subset $\mathcal{S} \subseteq \{1, \ldots, \frac{k}{r}\}$ of size $|\mathcal{S}| = k$, it is possible to reconstruct f_i from the encoded subfiles $\{\tilde{f}_i^{(s)}\}_{s \in \mathcal{S}}$.

Before the actual communication starts, each node prefetches some of these encoded subfiles. Specifically, any given encoded *subfile* is stored at any given node with probability $\frac{cr}{m}$, for a given $c > 0$, independently of whether other subfiles are stored at this node. Let $\pi_{n,i}$ denote the number of different subfiles of file f_i that node $n \in \mathcal{N}$ stores in its cache memory:

$$\pi_{n,i} := \left| \left\{ \ell \colon \tilde{f}_i^{\ell} \ \text{ stored in node } n\text{'s cache memory} \right\} \right|. \tag{2}$$

Notice that for each $i \in \{1, \ldots, m\}$:

$$\mathbb{E}[\pi_{n,i}] = \frac{ck}{m} =: \pi, \tag{3}$$

and by weak law of large numbers, with high probability, the number of effectively stored subfiles $\{\tilde{f}_i^{\ell}\}$ is close to this expected value. The parameter c is thus determined by the size of the cache memories at the nodes and it is assumed that all nodes have cache memories of same size.

Each node $n \in \mathcal{N}$ can harvest a fixed amount of energy $\mathsf{E}_0 > 0$ at the time-instances indicated by a corresponding stochastic counting process $\{Q_j^{\mathsf{e}}(n)\}_{j=1}^{\infty}$ of rate $\lambda_e > 0$. The energy harvesting processes at different nodes are assumed independent. Nodes can store the harvested energy in their batteries, which for simplicity are assumed have infinite capacity [12], and they can retrieve and consume this energy at any desired future point in time. Let $B_n(t)$ denote node n's battery level at time t. We have

$$B_n(t) := \mathsf{E}_0 \cdot \max\left\{j' > 0 \colon Q_{j'}^{\mathsf{e}}(n) \leq t\right\} - V_n(t), \tag{4}$$

where $V_n(t)$ is the energy consumed by node n prior to t.

A. File Delivery Process

Communication is governed by a centralized stochastic counting process $\{Q_j^{\mathsf{r}}\}_{j=1}^{\infty}$ of rate $\lambda_{\mathsf{r}} > 0$ and Q_j^{r} over the interval $[0, \mathsf{T}_{\text{max}}]$. We denote this process as the *request process*. At any given request time Q_j^{r}, the following three-steps protocol is carried out:

1.) File request: A demanding node N_j is selected uniformly at random from the set of nodes $\mathcal{N}$. The selected node N_j then produces a demand D_j uniformly at random over $\{1, \ldots, m\}$ to indicate that it wishes to learn file f_{D_j} and broadcasts this demand D_j to the set of nodes $\mathcal{N}_j$ that lie within a distance of at most $d_{\max}$ from node N_j. As we explain shortly, depending on the chosen strategy, the selected node N_j can also exchange additional information with the nodes in its neighbourhood $\mathcal{N}_j$. Note that at each request time only one file demand is addressed.

2.) Response: Nodes in the neighbourhood $\mathcal{N}_j$ respond to request Q_j^{r} by sending some of the encoded subfiles in their cache memories to the demanding node N_j. The same power P_0 and duration T_0 are used for each of the transmissions and at all nodes. The choice of these parameters as well as the strategies on how to select the encoded subfiles to send are design parameters. In Section III, we compare different file selection strategies. As we explain in Subsection II-B, the transmissions are not error-free.

3.) File reconstruction: The goal is that after receiving all the responses, the requesting node N_j has learned $k - \pi$ new encoded subfiles $\tilde{f}_{D_j}$ that are not yet stored in its cache memory. With the encoded subfiles from the cache memory, it can then reconstruct the desired file f_{D_j}. If this is not the case, we say that an error occurs and the system is in outage.

It is assumed that a new request starts only after the previous one ended. There are no collisions between different requests.

B. Channel Model

The various nodes that respond to a given request are assumed to communicate over different frequency bands and thus do not interfere. At any given time t, the communication from a node i to a node j is a fading channel, i.e., for input signal $x_{i,j}(t)$ at node i, node j observes the output signal

$$y_{i,j}(t) = h_{i,j}(t) x_{i,j}(t) + z_{i,j}(t), \quad t > 0, \tag{5}$$

---

[1] Here $r \in (0, 1]$ refers to the rate of the MDS code.

where $h_{i,j}(t)$ denotes the fading coefficient and $z_{i,j}(t)$ is the additive white Gaussian noise process with zero mean and variance $\sigma^2 > 0$. For given transmit power P_0, the received signal-to-noise ratio (SNR) at node j is then described as:

$$\gamma_{i,j} = \frac{P_0|h(t)|^2}{\sigma^2 r_{i,j}^{\alpha}}, \tag{6}$$

where $r_{i,j}$ is the distance between node i and node j and α denotes a given path loss exponent that is assumed constant for all transmissions.

C. Modulation and Packet Error Probability

All transmitters employ M-ary pulse amplitude modulation (M-PAM) of power P_0 and sampling rate $1/T_0$, where the real number $P_0 > 0$ and positive integer M are design parameters. The symbol duration T_0 is assumed to be given (determined by the allocated frequency bandwidth) and much smaller than the time between requests so as to ensure that transmissions corresponding to different requests do not overlap.

For a fixed receive SNR $\gamma_{i,j}$, the bit error probability $P_{\text{error}}(\gamma_{i,j})$ for the transmission from node i to node j is thus given by [14, Chapter 5]:

$$P_{\text{error}}(\gamma_{i,j}) = \frac{2(M-1)}{M\log M}\mathbb{Q}\left(\sqrt{\frac{6\gamma_{i,j}\log M}{(M^2-1)}}\right), \tag{7}$$

where $\mathbb{Q}\left(x\right) \triangleq \frac{1}{\sqrt{2\pi}}\int_x^{\infty} e^{-u^2/2}du$ is the Gaussian $\mathbb{Q}$–function.

Depending on the chosen subfile selection strategies, multiple nodes send the same subfile as a response to the j-th request. An encoded subfile $\tilde{f}_{D_j}^{(s)}$ is received successfully by demanding node N_j, if at least one of the transmissions of this encoded subfile was successful (i.e., *not* in error). In our model transmissions from various nodes are assumed independent.

D. File Outage

An outage occurs for the j-th request if, at the end of the communication session, node N_j does not have k encoded subfiles pertaining to file f_{D_j}, and therefore cannot reconstruct its desired file. Depending on the file selection strategy, an outage can occur for any of the following reasons:

- *Cache Outage:* Less than k distinct encoded subfiles of f_{D_j} are stored in the cache memories of all the nodes in $\mathcal{N}_j$ or at node N_j. That is, even if the requesting node N_j was given access to all cache memories in $\mathcal{N}_j$, it could not reconstruct the desired file f_{D_j}.
- *Selection Outage:* The set of encoded subfiles selected for transmission contains less than $k-\pi$ distinct subfiles pertaining to f_{D_j} that are not present in the cache memory of requesting node N_j.
- *Energy Outage:* All nodes in the neighborhood $\mathcal{N}_j$ ran out of energy during the protocol.
- *Transmission Outage:* The transmission of more than $k-\pi$ new encoded subfiles was scheduled and all scheduled transmissions could be performed, but the requesting node has acquired fewer than $k-\pi$ new encoded subfiles.

We next present different selection strategies for the D2D transmissions in the delivery phase.

III. SUBFILE SELECTION STRATEGIES

In this section, we propose different strategies for the selection of the encoded subfiles to be transmitted.

A. No-Selection Strategy

Consider the j-th request. In this strategy, the requesting node only sends the demand index D_j to the neighbouring nodes, but no other information e.g., cache contents, is exchanged. Each node in $\mathcal{N}_j$, sends as many encoded subfiles as allowed by its current energy level. It picks these encoded subfiles uniformly at random over all subfiles pertaining to f_{D_j} that are stored in its cache memory. Each encoded subfile is transmitted at most once by the same node.

The strategy requires no synchronization and can be implemented even when nodes are not willing to share information about their cache contents due to security or privacy reasons.

B. Sequential Strategy — Fair Approach

As in the previous strategy, the requesting node only sends the demand index D_j. Each node in $\mathcal{N}_j$ shares its location with N_j.

The requesting node N_j orders all the nodes in $\mathcal{N}_j$ in increasing distance. Then, N_j asks in a round robin manner *a single* encoded subfile from each node, starting with the closest node, the second closest node, etc, until it has received $k-\pi$ new encoded subfiles *without error*. The requesting node does not specify which encoded subfile a given node should send, but simply that it should send one. At its turn, a given node n selects a subfile uniformly at random among all the encoded subfiles pertaining to the requested file f_{D_j} that are stored in its cache memory and that it has not send previously during this request. It sends this subfile, if its energy level permits it, i.e., if

$$B_n(Q_j^{\text{r}}) \geq P_0 T_0. \tag{8}$$

The requesting node thus adaptively decides whether to continue polling nodes or not, depending on how many new encoded subfiles it has previously received without error.

Transmission is stopped and results in an outage if each node has been polled π times or if all nodes ran out of energy.

C. Sequential Strategy — Greedy Approach

The strategy is similar to the previous one. The requesting node again polls the neighbouring nodes from the closest to the farthest, but each node is polled only once. Moreover, a given node n sends *as many* of the encoded subfiles pertaining to f_{D_j} stored in its cache memory *as permitted by its current energy level*. The requesting node stops polling nodes once it has received $k-\pi$ new encoded subfiles *without error*.

Transmission is stopped and results in an outage if the requesting node has polled all neighbouring nodes in $\mathcal{N}_j$ and has received less than $k-\pi$ new encoded subfiles without error.

D. Coordinated Strategy — Greedy Approach

Consider the j-th request. The requesting node N_j only sends the demand index D_j. Each node $n \in \mathcal{N}_j$ sends back the indices of the encoded subfiles pertaining to f_{D_j} that it has stored in its cache memory, as well as the maximum number of subfiles that it can transmit with its current battery level:

$$\beta_n(Q_j^{\mathrm{r}}) := \frac{B_n(Q_j^{\mathrm{r}})}{P_0 T_0}. \tag{9}$$

The requesting node N_j evaluates all these responses and tells each of the nodes in $\mathcal{N}_j$ exactly which encoded subfiles it should send. Node N_j requests in total $b(k-\pi)$ encoded subfiles, where it ensures that none of them is stored in its cache memory and that at least $k-\pi$ of the requested subfiles are different. Here, the positive integer b is a design parameter of the strategy and describes the excess factor of number of requests. (Recall that node N_j needs to learn at least $k-\pi$ new encoded subfiles to be able to reconstruct f_{D_j}.)

In this approach, node N_j sends its request messages for all nodes at once. The number of subfiles requested from each node is decided following the greedy approach. More specifically, N_j considers nodes from the closest to the farthest, and assigns for each node the maximum number of subfiles permitted by their current battery level. It continues until $b(k-\pi)$ subfiles are assigned and transmits request messages to the concerned nodes.

Transmission terminates once node N_j receives $b(k-\pi)$ encoded subfiles *with or without error*. An outage occurs if it has received less than $k-\pi$ new encoded subfiles *without error* and thus cannot reconstruct file f_{D_j}.

Note that unlike in the previous sequential strategy, here nodes only send encoded subfiles that are not stored in the cache memory of the requesting node N_j. Another important difference concerns the stopping time which is fixed and cannot be chosen depending on whether subfiles are decoded in error or not.

E. Coordinated Strategy — Fair Approach

This strategy is similar to the previous strategy, node N_j sends its request messages for the $b(k-\pi)$ subfiles at once, but requests are balanced across the nodes as much as possible. Specifically, node N_j defines its request messages in the same way as before, it considers nodes from the closest to the farthest, but now it starts assigning a single encoded subfile for each node. If the node that is farthest away has been assigned one subfile and the number of assigned subfiles is still less than $b(k-\pi)$, it restarts assigning a second subfile to nodes from the closest to the furthest.

As in the previous strategy, an outage occurs if it has received less than $k-\pi$ new encoded subfiles *without error*.

F. Adaptive Strategy — Fair or Greedy Approaches

These two strategies combine the sequential nature of the first two strategies with the subfile-selection feature of the coordinated strategies. That means, the requesting node again learns all the energy levels and the cache contents pertaining to the demanded file f_{D_j}. The exact subfile request for a given node are performed in a sequential way, in that requests depend on whether the transmissions of previously sent subfiles have resulted in an error or not. Specifically, node N_j polls nodes in the order of increasing distance, and at each poll it asks the node to send encoded subfiles that are not in its cache memory and that it has not yet received without error.

In the greedy approach each polled node sends as many missing encoded subfiles as permitted by its energy level. In the fair approach, it sends a single missing encoded subfile. In the fair approach, if there is more than one subfile in the cache memory of the polled node that is not yet known to the requesting node N_j, then N_j requests one of them at random. Polls are performed in a round robin manner over up to π rounds.

In the greedy approach, transmission thus stops after node N_j has received $k-\pi$ new encoded subfiles *without error*; after it has polled all the nodes in its neighborhood $\mathcal{N}_j$; or when all nodes in $\mathcal{N}_j$ run out of energy. An outage occurs in the latter two cases.

In the fair approach, transmission stops as well if node N_j has received $k-\pi$ new encoded subfiles *without error*; if it has polled all the nodes in its neighborhood π times; or if all nodes run out of energy. An outage occurs in the latter two cases.

G. Communication Complexity

In each subfile selection strategy, before the actual subfile transmission starts, the requesting node N_j needs to exchange information with the neighbouring nodes in $\mathcal{N}_j$. This information is gathered in different communication steps as explained in Table I for the various subfile selection strategies. Table II shows the number of bits required to perform each communication step. In the following we explain each communication step in more detail.

- The requesting node N_j broadcasts the demand index D_j. This step is required in every proposed subfile selection strategy and the number of bits required to perform this step is
$$\lceil \log m \rceil + \lceil \log(|\mathcal{N}_j| + 1) \rceil, \tag{10}$$
where m is the total number of files in the library and $\lceil \log(|\mathcal{N}_j|+1) \rceil$ is the number of bits required to transmit the identity of the requesting node N_j.
- Nodes in $\mathcal{N}_j$ send their locations to the requesting node N_j. This communication step is performed only in the sequential strategies and the number of bits required to make this step is
$$|\mathcal{N}_j|(2\lceil \log(|\mathcal{N}_j| + 1) \rceil + G), \tag{11}$$
where G is the number of bits required to send the location of a node, and $2\lceil \log |\mathcal{N}_j| \rceil$ is the number of bits required to specify the identities of the transmitting and the receiving nodes.
- Each node $n \in \mathcal{N}_j$ sends back the indices of the encoded subfiles pertaining to f_{D_j} that it has stored in its cache

TABLE I: Communication complexity in each category of subfile selection strategy.

Strategy Communication Complexity	No-Selection	Sequential	Coordinated	Adaptive
Requesting node N_j sends the demand index D_j to the neighbouring nodes	✓	✓	✓	✓
Nodes in $\mathcal{N}_j$ send their location to N_j	×	✓	×	×
Nodes in $\mathcal{N}_j$ send the stored subfile's indices and their energy level to N_j	×	×	✓	✓
Requesting node N_j tells nodes in $\mathcal{N}_j$ the number of subfiles to send	×	✓	×	×
Requesting node N_j tells nodes in $\mathcal{N}_j$ which subfile to send	×	×	✓	✓

TABLE II: Number of bits required to perform each communication step.

Communication Step	Number of Bits
Requesting node N_j sends the demand index D_j to the neighbouring nodes	$\lceil\log m\rceil + \lceil\log(\lvert\mathcal{N}_j\rvert + 1)\rceil$
Nodes in $\mathcal{N}_j$ send their location to N_j	$\lvert\mathcal{N}_j\rvert(2\lceil\log(\lvert\mathcal{N}_j\rvert + 1)\rceil + G)$
Nodes in $\mathcal{N}_j$ send the stored subfile's indices and their energy level to N_j	$\lvert\mathcal{N}_j\rvert\left(2\lceil\log(\lvert\mathcal{N}_j\rvert + 1)\rceil + \pi\left\lceil\log\frac{k}{r}\right\rceil + \lceil\log\pi\rceil\right)$
Requesting node N_j tells nodes in $\mathcal{N}_j$ the number of subfiles to send	$2a_f\lceil\log(\lvert\mathcal{N}_j\rvert + 1)\rceil$
Requesting node N_j tells nodes in $\mathcal{N}_j$ which subfile to send	$a_2\left(2\lceil\log(\lvert\mathcal{N}_j\rvert + 1)\rceil + a_3\lceil\log\pi\rceil\right)$

memory, as well as the maximum number of subfiles that it can transmit. This step is performed in the coordinated and the adaptive strategies and the number of bits required to make this step is

$$|\mathcal{N}_j|\left(2\lceil\log(|\mathcal{N}_j|+1)\rceil+\pi\left\lceil\log\frac{k}{r}\right\rceil+\lceil\log\pi\rceil\right), \quad (12)$$

where $\pi\lceil\log\frac{k}{r}\rceil$ is the number of bits required to transmit the subfiles' indices, and $\lceil\log\pi\rceil$ is the number of bits required to transmit the maximum number of subfile that a node can transmit based on its energy level.

- Node N_j asks a specific number of subfiles from each node in $\mathcal{N}_j$ but does not specify which encoded subfile a given node should send. This step is performed only in the sequential strategies and the number of bits required to make this step is

$$2a_f\lceil\log(|\mathcal{N}_j|+1)\rceil, \quad (13)$$

where a_f is the number of nodes that node N_j polls. In the fair approach a_f takes value from the set $\{k-\pi,\dots\pi|\mathcal{N}_j|\}$ and in the greedy approach takes value from the set $\{\frac{(k-\pi)}{\pi},\dots,|\mathcal{N}_j|\}$.

- Node N_j tells each of the nodes in $\mathcal{N}_j$ which encoded subfiles it should send. This step is performed in the coordinated and the adaptive strategies and the number of bits required to make this step in the fair approach is

$$a_1\left(2\lceil\log(|\mathcal{N}_j|+1)\rceil+\lceil\log\pi\rceil\right), \quad (14)$$

where a_1 is the number of nodes that node N_j polls and is equal to $b(k-\pi)$ in the coordinated-fair approach and takes value from the set $\{k-\pi,\dots,\pi|\mathcal{N}_j|\}$ in the adaptive-fair approach. In the greedy approach, this step requires sending

$$a_2\left(2\lceil\log(|\mathcal{N}_j|+1)\rceil+a_3\lceil\log\pi\rceil\right) \quad (15)$$

bits, where a_2 is the number of nodes that node N_j polls and takes value from the set $\{\frac{b(k-\pi)}{\pi}\dots,b(k-\pi)\}$ in the coordinated-greedy approach and takes value from the set $\{\frac{(k-\pi)}{\pi},\dots,|\mathcal{N}_j|\}$ in the adaptive-greedy approach. Parameter a_3 determines the number of subfiles that a node is asked to transmit and takes value in $\{1,\dots,\pi\}$.

IV. SIMULATION RESULTS

We consider nodes that are distributed in a square area of 250 m^2. There are $m = 10$ files in the library, each divided into $k = 10$ subfiles. Subfiles are then encoded using a (10,20) MDS-code with rate $r = 0.5$. Each node can store 20 subfiles in its cache memory. It thus randomly selects $c = 2$ encoded subfiles of each file and stores them in its memory.

The total duration of the communication is $\mathsf{T}_{\max} = 10^4$ seconds and the transmission time of each encoded subfile T_0 is set to 1 ms. Communication is over an AWGN flat fading channel with path loss exponent $\alpha = 4$. At each energy arrival, a node harvests $E_0 = 1$mJ of energy.

Figure 2 compares the outage probabilities of the coordinated–fair approach in function of the transmit power P_0 for different values of the parameter b. The optimal transmit power P_0 seems to be between 0.1W and 1.2W, and in this regime the outage probability is decreasing for increasing values of b. The reason is that often the transmission or selection outages are the limiting events. Rarely, the file delivery is incomplete because nodes run out of energy. The figure further shows the adaptive-fair approach. It always performs better than the coordinated–fair approach because the requests are sequential and subfiles that are received without error at node N_j are never selected again.

Figure 3 illustrates the outage probability of the different strategies in function of the transmit power P_0. We observe that most strategies perform best for P_0 around 1W. For lower powers there are too many communication errors and for larger powers energy outages are too frequent. This happens in particular for the no-selection and the sequential strategies where nodes select their transmitted subfiles uniformly at random over all the files in their cache memories. In general, the adaptive–greedy approach performs best. In fact, it combines the advantages of all other strategies: similar to the sequential strategies transmission stops after the requesting node has received $k-\pi$ new encoded subfiles *without error*, and similar to the coordinated strategies, the requesting node tells the other

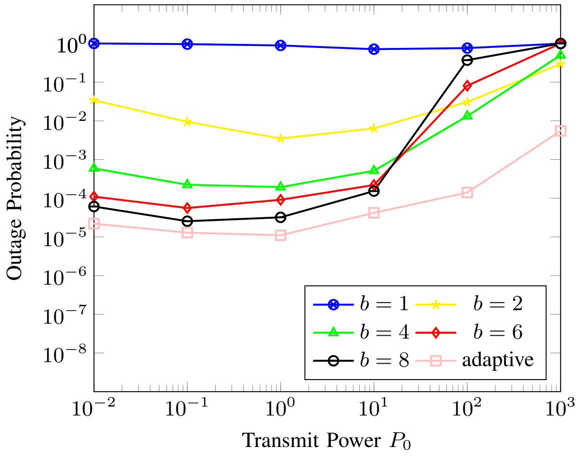

Fig. 2: Outage probability for coordinated–fair approach with different values of b. Here, $\lambda = 1, \lambda_e = 0.1, d_{\max} = 5, \lambda_r = 1$.

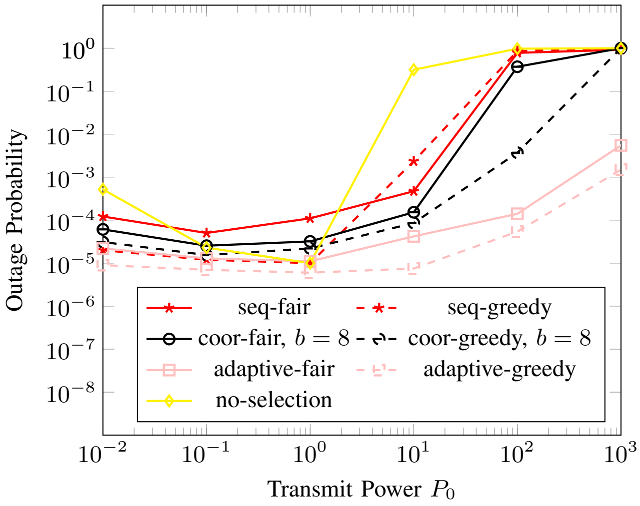

Fig. 3: Outage probability for different strategies in function of transmit power P_0. Here, $\lambda = 1, \lambda_e = 0.1, d_{\max} = 5, \lambda_r = 1$.

nodes exactly which files to send. The sequential–fair approach performs worse than the others for almost all power values. At powers around $P_0 = 1W$, most other strategies perform well, and there is thus not much reason to run the much more complicated adaptive–greedy approach. As we see in Figures 4–5, this conclusion does not hold for all parameter ranges.

Figure 4 shows the outage probability of our strategies in function of the request rate λ_r under fixed node density λ and energy arrival rate λ_e. Similarly to before, we observe a high (energy-)outage probability for the sequential and the no-selection strategies when only little energy is available *per request*, i.e., in the regime of high request rates $\lambda_r > 1$. The coordinated and the adaptive strategies are more robust and better manage the available energy by having node N_j exactly telling the neighbouring nodes in $\mathcal{N}_j$ which encoded subfiles to transmit. For Figure 4, we also observe that the fair approaches perform worse than the greedy approaches for small request rate, but become better as the rate of requests increases. The reason is that under the greedy approaches that demand most of the subfiles from close nodes, these nodes quickly run out of energy at high request rates. For low request rates nodes that have sent many subfiles have time to recover, i.e., harvest enough energy, before they are involved in the next request.

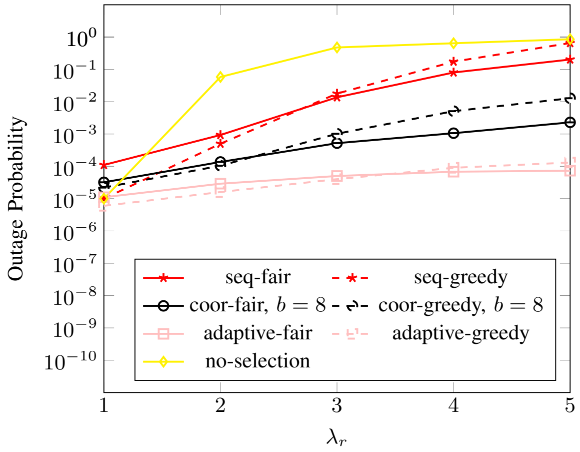

Fig. 4: Outage probability in function of the request rate λ_r. Here, $M = 2$, $P_0 = 1, \lambda_e = 0.1, d_{\max} = 5, \lambda = 1$.

Figure 5 reinforces the previous observations. It shows the outage probability of different strategies in function of the energy arrival rate λ_e when the node density λ and the request density λ_r are fixed. When the amount of energy in the system is moderate or small, then the coordinated and adaptive strategies perform much better than the no-selection or the sequential strategies. In fact, in these latter strategies, energy-outage events are frequent. For systems with lots of energy, i.e., when λ_e is large, then the no-selection and the sequential strategies can outperform the coordinated and adaptive strategies. Here, it is more important to avoid subfile selection outages.

Figure 6 shows the outage probability of our strategies in function of the node density λ under fixed request rate λ_r and energy arrival rate λ_e. Increasing the density of nodes while keeping the density of requests constant, is equivalent to adding more energy to the system. Higher energy in the system reduces the need for sophisticated and complex strategies by decreasing the energy outage probability. Therefore, less complex strategies such as the sequential and the no-selection strategies can perform as good as the adaptive and the coordinated strategies when the energy is high enough.

Figure 7 compares the communication complexity of our strategies in function of the number of neighbouring nodes in $\mathcal{N}_j$, which address a file demand. Note that $|\mathcal{N}_j|$ is a random variable and different strategies are compared for a fixed realization of $|\mathcal{N}_j|$. These communication steps have to be performed before the actual subfile transmission starts and as can be seen from this figure, the coordinated and the adaptive strategies are the most complex ones and the no-selection approach has the smallest communication complexity.

To summarize, we observed that when the amount of harvested energy in the network is low or when the density of

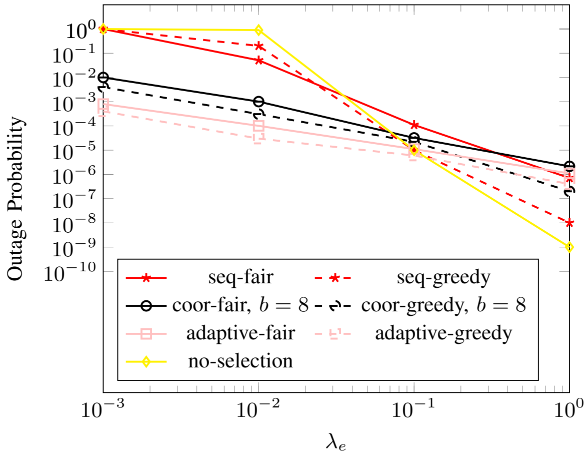

Fig. 5: Outage probability in function of the energy arrival rate λ_e. Here, $M = 2$ and $P_0 = 1, \lambda_r = 1, d_{\max} = 5, \lambda = 1$.

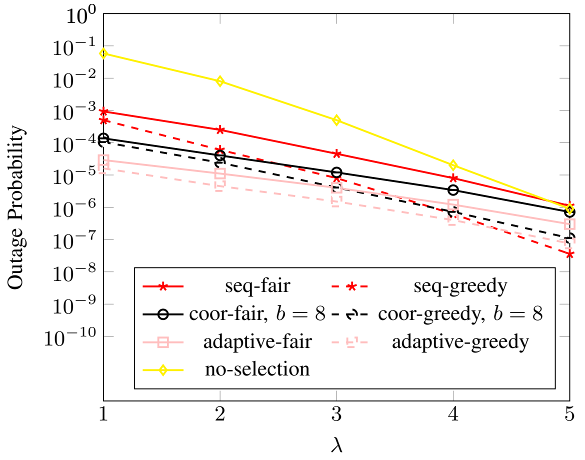

Fig. 6: Outage probability in function of the node density λ. Here, $M = 2$, $P_0 = 1, \lambda_e = 0.1, d_{\max} = 5, \lambda_r = 2$.

file requests is high, the proposed adaptive strategies should be employed. In fact, since in adaptive and coordinated strategies the demanding node exactly tells each of the neighbors which encoded subfiles to transmit, there is a lower chance that nodes run out of energy. In networks with high energy levels, sequential and no-selection strategies also show good performances and should be preferred due to their smaller complexity and delays. In general greedy strategies perform better when there is abundant energy in the system, and fair strategies perform better when energy is a scarce ressource.

V. CONCLUSIONS

The paper considers a stochastic wireless network with energy harvesting nodes that are equipped with cache memories and where the nodes' requests are satisfied through D2D communications from other nodes. The paper analyzes different selection strategies, i.e., different strategies to determine which neighbouring node should send which subfiles. Simulation results showed that a smart selection strategy that considers the nodes' energy levels and their distances can significantly improve performance over a naive strategy. Particularly when energy is a scarce resource in the network (e.g., due to a high request rate or small energy arrival rate), the proposed coordinated and adaptive strategies perform well. In these strategies the requesting node centrally decides which node should send which subfile, after being informed about the neighbouring nodes' cache contents. Moreover, in the adaptive strategies the requesting node takes the decision in a sequential manner so as to avoid that subfiles that have already been decoded successfully are retransmitted again. When the energy level is generally high, then it suffices to apply the simpler (less synchronization overhead and delay) sequential or no-selection strategies. In the sequential strategies, each node locally decides which files to transmit, but transmission stops as soon as the requesting node has obtained enough subfiles. In the no-selection strategy each node simply sends the maximum number of subfiles permitted by its energy level. The simpler strategies suffice also for high quality channels.

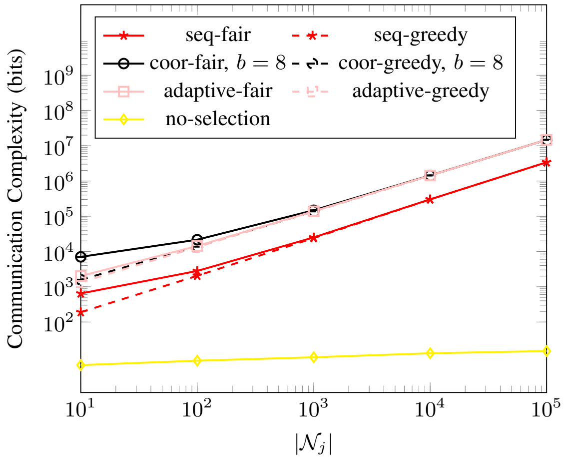

Fig. 7: Lower bound on the communication complexity of each strategy in function of $|\mathcal{N}_j|$. Here, $k = 100$ and $G = 10$.

ACKNOWLEDGEMENTS

H. Nikbakht, S. Kamel, and M. Wigger acknowledge funding from the ERC under grant agreement 715111. A. Yener acknowledges National Science Foundation Grant CCF-1749665. M. Wigger also acknowledges helpful discussions with P. Ciblat.

REFERENCES

[1] S. Ulukus, A. Yener, E. Erkip, O. Simeone, M. Zorzi, P. Grover, and K. Huang, "Energy harvesting wireless communications: A review of recent advances," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 360–381, Mar. 2015.

[2] K. Tutuncuoglu and A. Yener, "Optimum transmission policies for battery limited energy harvesting nodes," *IEEE Trans. Wireless Commun.*, vol. 11, no. 3, pp. 1180–1189, Mar. 2012.

[3] O. Ozel, K. Tutuncuoglu, J. Yang, S. Ulukus, and A. Yener, "Transmission with energy harvesting nodes in fading wireless channels: Optimal policies," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 8, pp. 1732–1743, Sep. 2011.

[4] K. Tutuncuoglu and A. Yener, "Sum-rate optimal power policies for energy harvesting transmitters in an interference channel," *JCN Special issue on Energy Harvesting in Wireless Networks*, vol. 14, no. 2, pp. 151–161, Apr. 2012.

[5] B. Varan and A. Yener, “Delay constrained energy harvesting networks with limited energy and data storage,” *IEEE J. Sel. Areas Commun.*, vol. 34,no. 5, pp. 1550–1564, May 2016.

[6] D. Shaviv and A. Ozgur, “Universally near optimal online power control for energy harvesting nodes,” *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3620– 3631, Dec. 2016.

[7] K. Tutuncuoglu, A. Yener, and S. Ulukus, “Optimum policies for an energy harvesting transmitter under energy storage losses,” *IEEE J. Sel. Areas Commun.*, vol. 33, no. 3, pp. 467–481, Mar. 2015.

[8] M. A. Maddah-Ali, and U. Niesen, “Fundamental limits of caching, in *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 28562867, May 2014.

[9] S. Zhou, J. Gong, Z. Zhou, W. Chen, and Z. Niu, “GreenDelivery: proactive content caching and push with energy-harvesting-based small cells,” *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 142–149, April 2015.

[10] A. Kumar, and W. Saad, “On the tradeoff between energy harvesting and caching in wireless networks,” in *Proc. IEEE ICCW 2015*, London, UK, June 8–12, 2015, pp. 1976–1981

[11] D. Niyato, D. I. Kim, P. Wang, and M. Bennis, “Joint admission control and content caching policy for energy harvesting access points,” *Proc. IEEE ICC 2016*, Kuala Lumpur, Malaysia, May 22–27, 2016, pp. 1–6

[12] J. Yang, and S. Ulukus, “Optimal packet scheduling in an energy harvesting communication system,” *IEEE Trans. Commun.*, vol. 60, no. 1, pp. 220–230, Jan. 2012.

[13] M. A. Maddah-Ali and U. Niesen, “Decentralized coded caching attains order-optimal memory-rate tradeoff, *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 10291040, Aug. 2015.

[14] A. Goldsmith, “Wireless Communications,” Cambridge, U.K.: Cambridge Univ. Press, 2004.